A complete algorithm to find exact minimal polynomial by approximations

Xiaolin Qin[a,b,c], Yong Feng[a,*], Jingwei Chen[a,b,c], Jingzhong Zhang[a,b]

[a] *Lab. of Computer Reasoning and Trustworthy Comput., University of Electronic Science and Technology of China, Chengdu 610054, China*
[b] *Lab. for Automated Reasoning and Programming, Chengdu Institute of Computer Applications, CAS, Chengdu 610041, China*
[c] *Graduate School of the Chinese Academy of Sciences, Beijing 100049, China*

Abstract

We present a complete algorithm for finding an exact minimal polynomial from its approximate value by using an improved parameterized integer relation construction method. Our result is superior to the existence of error controlling on obtaining an exact rational number from its approximation. The algorithm is applicable for finding exact minimal polynomial of an algebraic number by its approximate root. This also enables us to provide an efficient method of converting the rational approximation representation to the minimal polynomial representation, and devise a simple algorithm to factor multivariate polynomials with rational coefficients.

Compared with the subsistent methods, our method combines advantage of high efficiency in numerical computation, and exact, stable results in symbolic computation. we also discuss some applications to some transcendental numbers by approximations. Moreover, the *Digits* of our algorithm is far less than the LLL-lattice basis reduction technique in theory. In this paper, we completely implement how to obtain exact results by numerical approximate computations.

☆This research was partially supported by the National Key Basic Research Project of China (2004CB318003), the Knowledge Innovation Program of the Chinese Academy of Sciences (KJCX2-YW-S02), and the National Natural Science Foundation of China (10771205).

*Corresponding author.

Email addresses: qinxl@casit.ac.cn (Xiaolin Qin), yongfeng@casit.ac.cn (Yong Feng)

Keywords: algebraic number, numerical approximate computation, symbolic-numerical computation, integer relation algorithm, minimal polynomial

1. Introduction

Symbolic computations are principally exact and stable. However, they have the disadvantage of intermediate expression swell. Numerical approximate computations can solve large and complex problems fast, whereas only give approximate results. The growing demand for speed, accuracy and reliability in mathematical computing has accelerated the process of blurring the distinction between two areas of research that were previously quite separate. Therefore, algorithms that combine ideas from symbolic and numeric computations have been of increasing interest in the recent two decades. Symbolic computations are for sake of speed by intermediate use of floating-point arithmetic. The work reported in [22, 10, 15, 11, 9] studied the recovery of approximate value from numerical intermediate results. A somewhat related topic is algorithms that obtain the exact factorization of an exact input polynomial by use of floating point arithmetic in a practically efficient technique [6]. In the meantime, symbolic methods are applied in the field of numerical computations for ill-conditioned problems [8, 5, 23]. The main goal of hybrid symbolic-numeric computation is to extend the domain of efficiently solvable problems. However, there is a gap between approximate computations and exact results[25].

We consider the following question: Suppose we are given an approximate root of an unknown polynomial with integral coefficients and a bound on the degree and size of the coefficients of the polynomial. Is it possible to infer the polynomial and its exact root? The question was raised by Manuel Blum in Theoretical Cryptography, and Jingzhong Zhang in Automated Reasoning, respectively. Kannan *et al* answered the question in [17]. However, their technique is based on the Lenstra-Lenstra-Lovasz(LLL) lattice reduction algorithm, which is quite unstable in numerical computations. In [16], Just *et al* presented an algorithm for finding an integer relation on n real numbers using the LLL-lattice basis reduction technique, which needed the high precision. The function *MinimalPolynomial* in *maple*, which finds minimal polynomial for an approximate root, was implemented using the same technique.

In this paper, we present a new algorithm for finding exact minimal polynomial and reconstructing the exact root by approximate value. Our algorithm is based on the improved parameterized integer relation construction algorithm PSLQ(τ), whose stability admits an efficient implementation with lower run times on average than the existing algorithms, and can be used to prove that relation bounds obtained from computer runs using it is numerically accurate. The other function *identify* in *maple* , which finds a closed form for a decimal approximation of a number, was implemented using the integer relation construction algorithm. However, the choice of *Digits* of approximate value is fairly arbitrary [4]. In contrast, we fully analyze numerical behavior of an approximate to exact value and give how many *Digits* of approximate value, which can be obtained exact results. The work is regard as a further research in [26]. We solve the problem, which can be described as follows:

Given an approximate value $\tilde{\alpha}$ at arbitrary accuracy of an unknown algebraic number, and we also know the upper bound degree n of the algebraic number and an upper bound N of its height on minimal polynomial in advance. The problem will be solved in two steps: First, we discuss how much the error ε should be, so that we can reconstruct the algebraic number α from its approximation $\tilde{\alpha}$ when it holds that $|\alpha - \tilde{\alpha}| < \varepsilon$. Of course, ε is a function in n and N. Second, we give an algorithm to compute the minimal polynomial of the algebraic number.

Based on our method, we are able to extend our results with the same techniques to transcendental numbers of the form $\sin^{-1}(\alpha)$, $\log(\alpha)$, etc., where α is algebraic. we also propose a simple polynomial-time algorithm to factor multivariate polynomials with rational coefficients, and provide a natural, efficient technique to the minimal polynomial representation. The basic idea is from [17]. However, we have greatly improved their efficiency.

The rest of this paper is organized as follows. Section 2 illustrates the improved parameterized integer relation construction algorithm. Section 3 discusses how to reconstruct minimal polynomial and some applications to some transcendental numbers by approximations. Section 4 gives some experimental results. The final section concludes this paper.

This paper is the final journal version of [21], which contains essentially the entire contents of this paper.

2. Preliminaries

In this section, we first give some notations, and a brief introduction on integer relation problems. Then an improved parameterized integer relation construction algorithm is also reviewed.

2.1. Notations

Throughout this paper, $\mathbf{Z}$ denotes the set of the integers, $\mathbf{Q}$ the set of the rationals, $\mathbf{R}$ the set of the reals, $\mathbb{O}(\mathbf{R}^n)$ the corresponding system of ordinary integers, $U(n-1,\mathbf{R})$ the group of unitary matrices over $\mathbf{R}$, $GL(n,\mathbb{O}(\mathbf{R}))$ the group of unimodular matrices with entries in the reals, col_iB the i-th column of the matrix B. The ring of polynomials with integral coefficients will be denoted $\mathbf{Z}[X]$. The *content* of a polynomial $p(X)$ in $\mathbf{Z}[X]$ is the greatest common divisor of its coefficients. A polynomial in $\mathbf{Z}[X]$ is *primitive* if its content is 1. A polynomial $p(X)$ has degree d if $p(X) = \sum_{i=0}^{d} p_i X^i$ with $p_d \neq 0$. We write $deg(p) = d$. The *length* $|p|$ of $p(X) = \sum_{i=0}^{d} p_i X^i$ is the Euclidean length of the vector $(p_0, p_1, \cdots, p_d)$; the *height* $|p|_\infty$ of $p(X)$ is the L_∞-norm of the vector$(p_0, p_1, \cdots, p_d)$, so $|p|_\infty = \max_{0\leq i\leq d} |p_i|$. An *algebraic number* is a root of a polynomial with integral coefficients. The minimal polynomial of an algebraic number α is the irreducible polynomial in $\mathbf{Z}[X]$ satisfied by α. The minimal polynomial is unique up to units in $\mathbf{Z}$. The *degree* and *height* of an algebraic number are the degree and height of its minimal polynomial, respectively.

2.2. Integer relation algorithm

There exists an integer relation amongst the numbers $x_1, x_2, \cdots, x_n$ if there are integers $a_1, a_2, \cdots, a_n$, not all zero, such that $\sum_{i=1}^{n} a_i x_i = 0$. For the vector $\mathbf{x} = [x_1, x_2, \cdots, x_n]^T$, the nonzero vector $\mathbf{a} = [a_1, a_2, \cdots, a_n] \in \mathbf{Z}^n$ is an integer relation for $\mathbf{x}$ if $\mathbf{a} \cdot \mathbf{x} = 0$.

In order to introduce the integer relation algorithm, we recall some useful definitions and theorems[14, 3]:

Definition 2.1. ($M_{\mathbf{x}}$) Assume $\mathbf{x} = [x_1, x_2, \cdots, x_n]^T \in \mathbf{R}^n$ has norm $|\mathbf{x}|$=1. Define $\mathbf{x}^\perp$ to be the set of all vectors in $\mathbf{R}^n$ orthogonal to $\mathbf{x}$. Let $\mathbb{O}(\mathbf{R}^n) \cap \mathbf{x}^\perp$be the discrete lattice of integral relations for $\mathbf{x}$. Define $M_{\mathbf{x}} > 0$ to be the smallest norm of any relation for $\mathbf{x}$ in this lattice.

Definition 2.2. ($H_{\mathbf{x}}$) Assume $\mathbf{x} = [x_1, x_2, \cdots, x_n]^T \in \mathbf{R}^n$ has norm $|\mathbf{x}|$=1. Furthermore, suppose that no coordinate entry of $\mathbf{x}$ is zero, i.e., $x_j \neq 0$ for

$1 \leq j \leq n$(otherwise $\mathbf{x}$ has an immediate and obvious integral relation). For $1 \leq j \leq n$ define the partial sums

$$s_j^2 = \sum_{j \leq k \leq n} x_k^2.$$

Given such a unit vector $\mathbf{x}$, define the $n \times (n-1)$ lower trapezoidal matrix $H_{\mathbf{x}} = (h_{i,j})$ by

$$h_{i,j} = \begin{cases} 0 & \text{if } 1 \leq i < j \leq n-1, \\ s_{i+1}/s_i & \text{if } 1 \leq i = j \leq n-1, \\ -x_i x_j/(s_j s_{j+1}) & \text{if } 1 \leq j < i \leq n. \end{cases}$$

Note that $h_{i,j}$ is scale invariant.

Definition 2.3. (Modified Hermite Reduction) Let H be a lower trapezoidal matrix, with $h_{i,j} = 0$ if $j > i$ and $h_{j,j} \neq 0$. Set $D = I_n$, define the matrix $D = (d_{i,j}) \in GL(n, \mathbb{O}(\mathbf{R}))$ recursively as follows: For i from 2 to n, and for j from i-1 to 1(step -1), set $q = nint(h_{i,j}/h_{j,j})$; then for k from 1 to j replace $h_{i,k}$ by $h_{i,k} - qh_{j,k}$, and for k from 1 to n replace $d_{i,k}$ by $d_{i,k} - qd_{j,k}$, where the function *nint* denotes a nearest integer function, e.g., $nint(t) = \lfloor t + 1/2 \rfloor$.

Theorem 2.4. *Let $\boldsymbol{x} \neq 0 \in \mathbf{R}^n$. Suppose that for any relation $\boldsymbol{m}$ of $\boldsymbol{x}$ and for any matrix $A \in GL(n, \mathbb{O}(\mathbf{R}))$ there exists a unitary matrix Q$\in$ U(n-1) such that $H = AH_xQ$ is lower trapezoidal and all of the diagonal elements of H satisfy $h_{j,j} \neq 0$. Then*

$$\frac{1}{\max_{1 \leq j \leq n-1} |h_{j,j}|} = \min_{1 \leq j \leq n-1} \frac{1}{|h_{j,j}|} \leq |\boldsymbol{m}|.$$

Proof. See Theorem 1 of [14]. □

Remark 2.5. The inequality of Theorem 2.4 offers an increasing lower bound on the size of any possible relation. Theorem 2.4 can be used with any algorithm that produces $GL(n, \mathbb{O}(\mathbf{R}))$ matrices. Any $GL(n, \mathbb{O}(\mathbf{R}))$ matrix A whatsoever can be put into Theorem 2.4.

Theorem 2.6. *Assume real numbers, $n \geq 2$, $\tau > 1$, $\gamma > \sqrt{4/3}$, and that $0 \neq \boldsymbol{x} \in \mathbf{R}^n$ has $\mathbb{O}(\mathbf{R})$ integer relations. Let $M_{\boldsymbol{x}}$ be the least norm of relations for $\boldsymbol{x}$. Then PSLQ(τ) will find some integer relation for $\boldsymbol{x}$ in no more than*

$$\binom{n}{2} \frac{log(\gamma^{n-1} M_{\boldsymbol{x}})}{log\tau}$$

iterations.

Proof. See Theorem 2 of [14]. □

Theorem 2.7. *Let $M_{\boldsymbol{x}}$ be the smallest possible norm of any relation for $\boldsymbol{x}$. Let $\boldsymbol{m}$ be any relation found by PSLQ(τ). For all $\gamma > \sqrt{4/3}$ for real vectors*

$$|\boldsymbol{m}| \leq \gamma^{n-2} M_{\boldsymbol{x}}.$$

Proof. See Theorem 3 of [14]. □

Remark 2.8. For n=2, Theorem 2.7 proves that any relation $0 \neq \mathbf{m} \in \mathbb{O}(\mathbf{R}^2)$ found has norm $|\mathbf{m}| = M_{\mathbf{x}}$. In other words, $PSLQ(\tau)$ finds a shortest relation. For real numbers this corresponds to the case of the Euclidean algorithm.

Based on these theorems as above, and if there exists a known error controlling ε, then an algorithm for obtaining the integer relation can be designed as follows:

Algorithm 2.9. Parameterized Integer Relation Construction
Input: a vector $\mathbf{x}$, the upper bound N on the height of minimal polynomial, and an error $\varepsilon > 0$;
Output: an integer relation $\mathbf{m}$.

Step 1: Set $i := 1$, $\mathbf{m} := \mathbf{0}$, $\tau > 2/\sqrt{3}$, and unitize the vector $\mathbf{x}$ to $\bar{\mathbf{x}}$;

Step 2: Set $H_{\bar{\mathbf{x}}}$ by definition 2.2;

Step 3: Produce matrix $D \in GL(n, \mathbb{O}(\mathbf{R}))$ using modified Hermite Reduction by definition 2.3;

Step 4: Set $\bar{\mathbf{x}} := \bar{\mathbf{x}} \cdot D^{-1}, H := D \cdot H, A := D \cdot A, B := B \cdot D^{-1}$,
case 1: if $\bar{\mathbf{x}}_j = 0$, then $\mathbf{m} := col_j B$;
case 2: if $h_{i,i} < \varepsilon$, then $\mathbf{m} := col_{n-1} B$;

Step 5: if $0 < |\mathbf{m}|_\infty \leq N$, then goto Step 12;
if $|\mathbf{m}|_\infty > N$, there is no such an integer relation, algorithm terminating.

Step 6: $i := i + 1$;

Step 7: Choose an integer r, such that $\tau^r |h_{r,r}| \geq \tau^j |h_{j,j}|$, for all $1 \leq j \leq n-1$;

Step 8: Define $\alpha := h_{r,r}$, $\beta := h_{r+1,r}$, $\lambda := h_{r+1,r+1}$, $\sigma := \sqrt{\beta^2 + \lambda^2}$;

Step 9: Change h_r to h_{r+1}, and define the permutation matrix R;

Step 10: Set $\bar{\mathbf{x}} := \bar{\mathbf{x}} \cdot R$, $H := R \cdot H$, $A := R \cdot A$, $B := B \cdot R$, if i=n-1, then goto Step 4;

Step 11: Define $Q := (q_{i,j}) \in U(n-1, \mathbf{R})$, $H := H \cdot Q$, goto Step 4;

Step 12: return $\mathbf{m}$.

By algorithm 2.9, we can find the integer relation $U(n-1, \mathbf{R})$ of the vector $\mathbf{x} = (1, \tilde{\alpha}, \tilde{\alpha}_2, \cdots, \tilde{\alpha}_n)$. So, we get a nonzero polynomial of degree n, which denotes $G(x)$ for the rest of this paper, i.e.,

$$G(x) = \mathbf{m} \cdot (1, x, x^2, \cdots, x^n)^T. \tag{2.1}$$

Our main task is to show that polynomial (2.1) is uniquely determined under assumptions, and discuss the controlling error ε in algorithm 2.9 in the next section.

3. Reconstructing minimal polynomial from its approximation

In this section, we will solve such a problem: For a given floating number $\tilde{\alpha}$, which is an approximation of unknown algebraic number α, how do we obtain the exact value? At first, we state some lemmas as follows:

Lemma 3.1. *Let f be a nonzero polynomial in $\mathbf{Z}[x]$ of degree n. If $\varepsilon = \max_{1 \le i \le n} |\alpha^i - \tilde{\alpha}_i|$[1], where $\tilde{\alpha}_i$ for $1 \le i \le n$ are the rational approximations to the powers α^i of an algebraic number α, and $\tilde{\alpha}_0 = 1$, then*

$$|f(\alpha) - f(\tilde{\alpha})| \le \varepsilon \cdot n \cdot |f|_\infty. \tag{3.1}$$

Proof. Clear. □

Lemma 3.2. *Let h and g be two nonzero polynomials in $\mathbf{Z}[x]$ of degree n and m, respectively, and let $\alpha \in \mathbf{R}$ be a zero of h with $|\alpha| \le 1$. If h is irreducible and $g(\alpha) \ne 0$, then*

$$|g(\alpha)| \ge n^{-1} \cdot |h|^{-m} \cdot |g|^{1-n}. \tag{3.2}$$

[1] ε is defined by the same way for the rest of this paper.

Proof. See Proposition(1.6) of [17]. If $|\alpha| > 1$, a simple transform of it does. □

Corollary 3.3. *Let h and g be two nonzero polynomials in $\mathbf{Z}[x]$ of degrees n and m, respectively, and let $\alpha \in \mathbf{R}$ be a zero of h with $|\alpha| \leq 1$. If h is irreducible and $g(\alpha) \neq 0$, then*

$$|g(\alpha)| \geq n^{-1} \cdot (n+1)^{-\frac{m}{2}} \cdot (m+1)^{\frac{1-n}{2}} \cdot |h|_\infty^{-m} \cdot |g|_\infty^{1-n}. \tag{3.3}$$

Proof. First notice that $|f|^2 \leq (n+1) \cdot |f|_\infty^2$ holds for any polynomial f of degree at most $n > 0$, so $|f| \leq \sqrt{n+1} \cdot |f|_\infty$. From Lemma 3.2, we get

$$|g(\alpha)| \geq n^{-1} \cdot (n+1)^{-\frac{m}{2}} \cdot (m+1)^{\frac{1-n}{2}} \cdot |h|_\infty^{-m} \cdot |g|_\infty^{1-n}.$$

This proves Corollary 3.3. □

Theorem 3.4. *Let $\tilde{\alpha}$ be an approximate value to an unknown algebraic number α with degree $n > 0$, N be the upper bound on the height of minimal polynomial of α. For any $G(x)$ in $\mathbf{Z}[x]$ with degree n, if*

$$|G(\tilde{\alpha})| < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n} - n \cdot \varepsilon \cdot |G|_\infty,$$

then

$$|G(\alpha)| < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n}.$$

Proof. Let $\alpha \in \mathbf{R}$ be with $|\alpha| \leq 1$. From Lemma 3.1, we notice that $|G(\alpha) - G(\tilde{\alpha})| \leq \varepsilon \cdot n \cdot |G|_\infty$, and

$$|G(\alpha)| - |G(\tilde{\alpha})| \leq |G(\alpha) - G(\tilde{\alpha})|,$$

so,

$$|G(\alpha)| \leq |G(\tilde{\alpha})| + n \cdot \varepsilon \cdot |G|_\infty. \tag{3.4}$$

From the assumption of the theorem, since

$$|G(\tilde{\alpha})| < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n} - n \cdot \varepsilon \cdot |G|_\infty, \tag{3.5}$$

combined with (3.4), we have proved Theorem 3.4. □

Corollary 3.5. *Let $\tilde{\alpha}$ be an approximate value to an unknown algebraic number α with degree $n > 0$, N be the upper bound on the height of minimal polynomial of α. For any $G(x)$ in $\mathbf{Z}[x]$ with degree n, if $|G(\alpha)| < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n}$, then*

$$G(\alpha) = 0. \tag{3.6}$$

The primitive part of polynomial $G(x)$ is the minimal polynomial of algebraic number α.

Proof. (Proof by Contradiction) Let $\alpha \in \mathbf{R}$ be with $|\alpha| \leq 1$. According to Lemma 3.2, suppose on the contrary that $G(\alpha) \neq 0$, then

$$|G(\alpha)| \geq n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n}.$$

From the assumption of the corollary, we have

$$|G(\alpha)| < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n}.$$

However, it leads to a contradiction. So, $G(\alpha) = 0$.
Let $G(x) = \sum_{i=0}^{n} a_i x^i$, which is constructed by the parameterized integer relation construction algorithm from the vector $\mathrm{x} = (1, \alpha, \alpha^2, \cdots, \alpha^n)$. Since algebraic number α with degree $n > 0$, according to the definition of minimal polynomial, then the primitive polynomial of $G(x)$, denoted by $pp(G(x))$. Hence $pp(G(x))$ is just irreducible and equal to $g(x)$. Of course, it is unique. This proves Corollary 3.5. □

3.1. Obtaining minimal polynomial by approximation

If α is a real number, then by definition α is algebraic exactly, for some n, the vector

$$(1, \alpha, \alpha^2, \cdots, \alpha^n) \tag{3.7}$$

has an integer relation. The integral coefficients polynomial of lowest degree, whose root an algebraic number α is, is determined uniquely up to a constant multiple; it is called the *minimal polynomial* for α. Integer relation algorithm can be employed to search for minimal polynomial in a straightforward way by simply feeding them the vector (3.7) as their input. Let $\tilde{\alpha}$ be an approximate value belonging to an unknown algebraic number α, considering the vector $\mathbf{v} = (1, \tilde{\alpha}, \tilde{\alpha}^2, \cdots, \tilde{\alpha}^n)$, how to obtain the exact minimal polynomial from its approximate value? We have the same technique answer to the question from the following theorem.

Theorem 3.6. *Let $\tilde{\alpha}$ be an approximate value belonging to an unknown algebraic number α of degree $n > 0$. If*

$$\varepsilon = |\alpha - \tilde{\alpha}| < 1/(n^2(n+1)^{n-\frac{1}{2}}N^{2n}), \tag{3.8}$$

where N is the upper bound on the height of its minimal polynomial, then $G(\alpha) = 0$, and the primitive part of $G(x)$ is its minimal polynomial.

Proof. Let $\alpha \in \mathbf{R}$ be with $|\alpha| \leq 1$. From Theorem 3.4 and Corollary 3.5, it is obvious that

$$G(\alpha) = 0,$$

if and only if

$$|G(\alpha)| < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n}. \tag{3.9}$$

Under the assumption of the theorem, we get the upper bound of degree n and an approximate value $\tilde{\alpha}$ belonging to an unknown algebraic number α. For substituting the approximate value $\tilde{\alpha}$ in $G(x)$, denoted by $G(\tilde{\alpha})$, there are two cases:
Case 1: $G(\tilde{\alpha}) \neq 0$, $|G(\tilde{\alpha})| > 0$. We have the inequality (3.5)holds, i.e.,

$$0 < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n} - n \cdot \varepsilon \cdot |G|_\infty. \tag{3.10}$$

Clearly, the inequality (3.10) satisfies from the condition (3.8). This proves the Case 1.
Case 2: $G(\tilde{\alpha}) = 0$. From Lemma 3.1, we have $|G(\alpha) - G(\tilde{\alpha})| < n \cdot \varepsilon \cdot |G|_\infty$, hence $|G(\alpha)| < n \cdot \varepsilon \cdot |G|_\infty$. In order to satisfy condition (3.9), we only need the following inequality holds,

$$n \cdot \varepsilon \cdot |G|_\infty < n^{-1} \cdot (n+1)^{-n+\frac{1}{2}} \cdot |G|_\infty^{-n} \cdot N^{1-n}. \tag{3.11}$$

From theorem 2.7, and algorithm 2.9 in Step 5, $|G|_\infty$ is not more than N. Hence we replace $|G|_\infty$ by N. So the correctness of the inequality (3.11) follows from (3.8). This proves Theorem 3.6. □

It is easiest to appreciate the theorem by seeing how it justifies the following algorithm for obtaining minimal polynomials from its approximation:

Algorithm 3.7. Reconstructing Minimal Polynomial
Input: a floating number ($\tilde{\alpha}$, n, N) belonging to an unknown algebraic number α, i.e., satisfying (3.8).
Output: $g(x)$, the minimal polynomial of α.

Step 1: Construct the vector v;

Step 2: Compute ε satisfying (3.8);

Step 3: Call algorithm 2.9 to find an integer relation w for v;

Step 4: Obtain $w(x)$ the corresponding polynomial;

Step 5: Evaluate the primitive part of $w(x)$ to $g(x)$;

Step 6: return $g(x)$.

Theorem 3.8. *Algorithm 3.7 works correctly as specified and uses $O(n(logn+logN))$ binary bit operations, where n and N are the degree and height of its minimal polynomial, respectively.*

Proof. Correctness follows from Theorem 3.6. The cost of the algorithm is $\mathrm{O}(n(logn + logN))$ binary bit operations obviously. □

3.2. Some applications

In this subsection, we discuss some applications to the practicalities. The method of obtaining exact minimal polynomial from an approximate root can be extended to the set of complex numbers and many applications in computer algebra and science.

This yields a simple factorization algorithm for multivariate polynomials with rational coefficients: We can reduce a multivariate polynomial to a bivariate polynomial using the Hilbert irreducibility theorem, the basic idea was described in [9], and then convert a bivariate polynomial to a univariate polynomial by substituting a transcendental number in [24] or an algebraic number of high degree for one variate in [7]. After this substitution we can get an approximate root of the univariate polynomial and use our algorithm to find the irreducible polynomial satisfied by the exact root, which must then be a factor of the given polynomial. It can find the bivariate polynomial's factors, from which the factors of the original multivariate polynomial can be recovered using Hensel lifting. This is repeated until all the factors are found.

The other yields an efficient method of converting the rational approximation representation to the minimal polynomial representation of an algebraic number. The traditional representation of algebraic numbers is by their minimal polynomials [1, 2, 13, 19]. We now propose an efficient method

to the minimal polynomial representation, which only needs an approximate value, degree and height of its minimal polynomial, i.e., an ordered triple $< \tilde{\alpha}, n, N >$ instead of an algebraic number α, where $\tilde{\alpha}$ is its approximate value, and n and N are the degree and height of its minimal polynomial, respectively, denoted by $< \alpha >=< \tilde{\alpha}, n, N >$. It is not hard to see that the computations in the representation can be changed to computations in the other without loss of efficiency, the rational approximation method is closer to the intuitive notion of computation.

we also discuss some applications to some transcendental numbers by using an improved parameterized integer relation construction method. The form of these transcendental numbers is $\sin^{-1}(\alpha)$, $\cos^{-1}(\alpha)$, $\log(\alpha)$ etc., where α is an algebraic number. Moreover, a large number of results were found by using integer relation detection algorithm in the course of research on multiple sums and quantum field theory in [12].

Suppose β is the principle value of $\sin^{-1}(\alpha)$ for some unknown α, which is, however, known to the algebraic of degree and height at most n and N, respectively. We consider inferring the minimal polynomial of α from an approximation $\tilde{\beta}$ to β in the deterministic polynomial time. We show that if $|\beta - \tilde{\beta}|$ is at most $\varepsilon = 1/(n^2(n+1)^{n-\frac{1}{2}}N^{2n})$, this can be done. The specific technique is similar with the method in [17].

Thus, in polynomial time we can compute from $\tilde{\beta}$ an approximation $\tilde{\alpha}$ to an unknown algebraic number α such that $|\alpha - \tilde{\alpha}| \leq \varepsilon$, with ε as above. Now Theorem (3.6) guarantees that we can find the minimal polynomial of α in polynomial time.

4. Experimental results

Our algorithms have been implemented in *Maple*. The following examples run in the platform of Maple 12 under Windows and PIV2.53GB, 512MB of main memory. Table 1 proposes the $Digits$ of approximate values to compare our method against the LLL-lattice basis reduction algorithm.

In Table 1, we present many examples to compare our new method against the LLL-lattice basis reduction algorithm. For each example, we construct the irreducible polynomial with random integral coefficients in the range $-20 \leq c \leq 20$. Here n and N denote the degree of algebraic number and the height of its corresponding minimal polynomial respectively; whereas D_{LLL} and D_{PSLQ} are relative $Digits$ to obtain the exact minimal polynomial in theory, E_{PSLQ} is in our optimal experimental results respectively.

$Ex.$	n	N	D_{LLL}	D_{PSLQ}	E_{PSLQ}
1	4	13	16	12	8
2	7	17	36	25	11
3	10	15	57	36	16
4	15	19	102	59	25
5	23	9	145	77	29
6	27	19	240	109	55
7	30	15	277	118	57
8	34	11	327	126	67
9	40	15	435	161	82
10	45	17	532	189	110
11	50	13	620	200	123
12	100	13	2033	427	381

Table 1: Comparison between our algorithm and LLL-lattice basis reduction technique

From Table 1, we observe that the *Digits* of our algorithm is far less than the LLL-lattice basis reduction technique in theory. However, the *Digits* of our algorithm is more than that of the optimal experimental results. So, in the further work we would like to consider improving the error controlling.

The following first two examples illuminate how to obtain an exact quadratic algebraic number and minimal polynomial. Example 3 uses a simple example to test our algorithm for factoring primitive polynomials with integral coefficients.

Example 4.1. Let α be an unknown quadratic algebraic number. We only know an upper bound of height on its minimal polynomial $N = 47$. According to theorem 3.6, compute quadratic algebraic number α as follows: First obtain control error $\varepsilon = 1/(12 * \sqrt{3} * N^4) = 1/(1807729447692 * \sqrt{3}) \approx 1.0 \times 10^{-8}$. And then assume that we use some numerical method to get an approximation $\tilde{\alpha} = 11.937253933$, such that $|\alpha - \tilde{\alpha}| < \varepsilon$. Calling algorithm 3.7 yields as follows: Its minimal polynomial is $g(x) = x^2 - 8 * x - 47$. So, we can obtain the corresponding quadratic algebraic number $\alpha = 4 + 3\sqrt{7}$.

Remark 4.2. The function *identify* in maple 12 needs *Digits*:=13, whereas our algorithm only needs 9 digits.

Example 4.3. Let a known floating number $\tilde{\alpha}$ belonging to some algebraic number α of degree $n = 4$, where $\tilde{\alpha} = 3.14626436994198$, we also know an

upper bound of height on its minimal polynomial $N = 10$. According to theorem 3.6, we can get the error $\varepsilon = 1/(n^2(n+1)^{n-\frac{1}{2}}N^{2n}) = 1/(4^2 * 5^{\frac{7}{2}} * 10^8) \approx 2.2 \times 10^{-12}$. Calling algorithm 3.7, if only the floating number $\tilde{\alpha}$, such that $|\alpha - \tilde{\alpha}| < \varepsilon$, then we can get its minimal polynomial $g(x) = x^4 - 10 * x^2 + 1$. So, the exact algebraic number α is able to be denoted by $< \alpha >=< 3.14626436994198, 4, 10 >$, i.e., $< \sqrt{2} + \sqrt{3} >=< 3.14626436994198, 4, 10 >$.

Remark 4.4. In the example 4.3, we only propose a simple example to represent the exact algebraic number by approximate method. In the further work, we would like to discuss the efficient arithmetic algorithms for summation, multiplication and inverse of the rational approximation representation.

Example 4.5. This example is an application in factoring primitive polynomials over integral coefficients. For convenience and space-saving purposes, we choose a very simple polynomial as follows:

$$p = 3x^9 - 9x^8 + 3x^7 + 6x^5 - 27x^4 + 21x^3 + 30x^2 - 21x + 3$$

We want to factor the polynomial p via reconstruction of minimal polynomials over the integers. First, we transform p to a primitive polynomial as follows:

$$p = x^9 - 3x^8 + x^7 + 2x^5 - 9x^4 + 7x^3 + 10x^2 - 7x + 1,$$

We see the upper bound of coefficients on polynomial p is 10, which has relation with an upper bound of coefficients of the factors on the primitive polynomial p by Landau-Mignotte bound [20]. Taking $N = 5$, $n = 2$ yields $\varepsilon = 1/(2^2 * (2+1)^{2-\frac{1}{2}} * 5^4) = 1/(7500 * \sqrt{3}) \approx 8.0 \times 10^{-5}$. Then we compute the approximate root on x. With Maple we get via [fsolve$(p = 0, x)$]:
$S = [2.618033989, 1.250523220, -.9223475138, .3819660113, .2192284350]$.

According to theorem 3.6, let $\tilde{\alpha} = 2.618033989$ be an approximate value belonging to some quadratic algebraic number α, calling algorithm 3.7 yields as follows:

$$p_1 = x^2 - 3 * x + 1.$$

And then we use the polynomial division to get

$$p_2 = x^7 + 2 * x^3 - 3 * x^2 - 4 * x + 1.$$

Based on the Eisenstein's Criterion [18], the p_2 is irreducible in $\mathbf{Z}[X]$. So, the p_1 and p_2 are the factors of primitive polynomial p.

5. Conclusions

In this paper, we propose a new method for obtaining exact results by numerical approximate computations. The key technique of our method is based on an improved parameterized integer relation construction algorithm, which is able to find an exact relation by the accuracy control. The error ε in formula (3.8) is an exponential function in degree and height of its minimal polynomial. The result is superior to the existence of error controlling on obtaining an exact rational number from its approximation in [26]. Using our algorithm, we have succeed in factoring multivariate polynomials with rational coefficients and providing an efficient method of converting the rational approximation representation to the minimal polynomial representation. Our method can be applied in many aspects, such as proving inequality statements and equality statements, and computing resultants, etc.. Thus we can take fully advantage of approximate methods to solve larger scale symbolic computation problems. Furthermore, our basic idea can be generalized easily to complex algebraic numbers.

[1] Boehm, H.J. and Cartwright, R. Exact real arithmetic: Formulating real numbers as functions. In Turner. D., editor, *Research Topics in Functional Programming,* Addison-Wesley, (1990), pp. 43-64.

[2] Boehm, H.J., Cartwright, R., Riggle, M., et al. Exact real arithmetic: A case study in higher order programming. *In ACM Symposium on Lisp and Functional Programming*, 1986.

[3] Borwein, J. M., and Lisonek, P. Applications of Integer Relation Algorithms. *Disc. Math.* 217(2000): 65-82.

[4] Borwein, P., Hare, K. G. and Meichsener, A. Reverse Symbolic Computations, The *identity* Function. in: Proceedings from the Maple Summer Workshop, Maple Software, Waterloo, 2002.

[5] Brown, W. S. On Euclid's algorithm and the computation of polynomial greatest common divisors. *J. ACM.* 18, 4(1971)): 478-504.

[6] Chèze, G., Galligo, A. From an approximate to an exact absolute polynomial factorization. *Journal of Symbolic Computation,* 41: 682-696, 2006.

[7] Chen, J. W., Feng Y., Qin X. L., et al. Exact Polynomial Factorization by Approximate High Degree Algebraic Numbers. In Proc. 2009 Internat. Workshop on Symbolic-Numeric Comput. ACM press, 21-28,2009.

[8] Collins, G. E. Polynomial remainder sequences and determinants. *Amer. Math. Monthly,* 73(1966): 708-712.

[9] Corless, R. M., Galligo, A., Kotsireas, I. S., et al. A geometric-numeric algorithm for absolute factorization of multivariate polynomials. *In Proc. 2002 Internat. Symp. Comput. ISSAC'02*, ACM press, pp. 37-45.

[10] Corless, R.M., Giesbrecht, M. W., et al. Numerical implicitization of parametric hypersurfaces with linear algebra. *In Proc. AISC2000, LNAI 1930*, pp. 174-183.

[11] Corless, R. M., Giesbrecht, M. W., et al. Towards factoring bivariate approximate polynomials. *In Proc. 2001 Internat. Symp. Comput. ISSAC'01*, ACM press, pp. 85-92.

[12] David H. B.. Integer Relation Detection. Computing in Science and Engineering, 2, 1(2000), pp. 24-28.

[13] Edalat, A. and Potts, P. J. A new representation for exact real numbers. *In Mathematical foundations of programming semantics (Pittsburgh, PA, 1997)*, page 14 pp. (electronic). Elsevier, Amsterdam, 1997.

[14] Ferguson, H. R. P., Bailey, D. H., and Arno, S. Analysis of PSLQ, An Integer Relation Finding Algorithm. *Math. Comput.*, 68, 225(1999): 351-369.

[15] Huang, Y., Wu, W., Stetter, H., et al. Pseudofactors of multivariate polynomials. *In Proc. 2000 Internat. Symp. Comput. ISSAC'00*, ACM press, pp. 161-168.

[16] Just, B. Integer relations among algebraic numbers. Mathematical Foundations of Computer Science, (1989), pp. 314-320.

[17] Kannan, R., Lenstra, A.K., and Lovász, L. Polynomial Factorization and Nonrandomness of Bits of Algebraic and Some Transcendental Numbers. *Math.Comput.* 50, 182(1988): 235-250.

[18] Lang, S. *Algebra*, 3rd edition, Springer-Verlag, New York, 2002.

[19] Loos, B. Computing in Algebraic Extensions, *Computer Algebra,* (Ed. by B. Buchberger, et al), Springer-Verlag, (1982), pp. 173-187.

[20] Mignotte, M., An inequality about factors of polynomials, *Math. Comp.*, 28, 128(1974): 1153-1157.

[21] Qin, X. L., Feng, Y., Chen J. W., et al. Finding Exact Minimal Polynomial by Approximations. In Proc. 2009 Internat. Workshop on Symbolic-Numeric Comput. ACM press,125-131,2009.

[22] Sasaki, T., Suzuki, M., et al. Approximate factorization of multivariate polynomials and absolute irreducibility testing. *Japan J. Indust. Appl. Math.* 8 (1991), 357-375.

[23] Schönhage, A. Quasi-gcd computations. *J.Complexity*. 1(1985): 118-137.

[24] Van Der Hulst, M.-P. and Lenstra, A. K. Factorization of polynomials by transcendental evaluation. *EUROCAL'85*, (1985), pp. 138-145.

[25] Yang, L., Zhang, J. Z., and Hou, X. R. A criterion of dependency between algebraic equation and its application. Proc. *IWMN'92,* Internat. Academic Publishers, pp. 110-134, 1992.

[26] Zhang, J. Z., Feng, Y. Obtaining Exact Value by Approximate Computations. *Science in China Series A: Mathematics*. 50, 9(2007): 1361-1368.